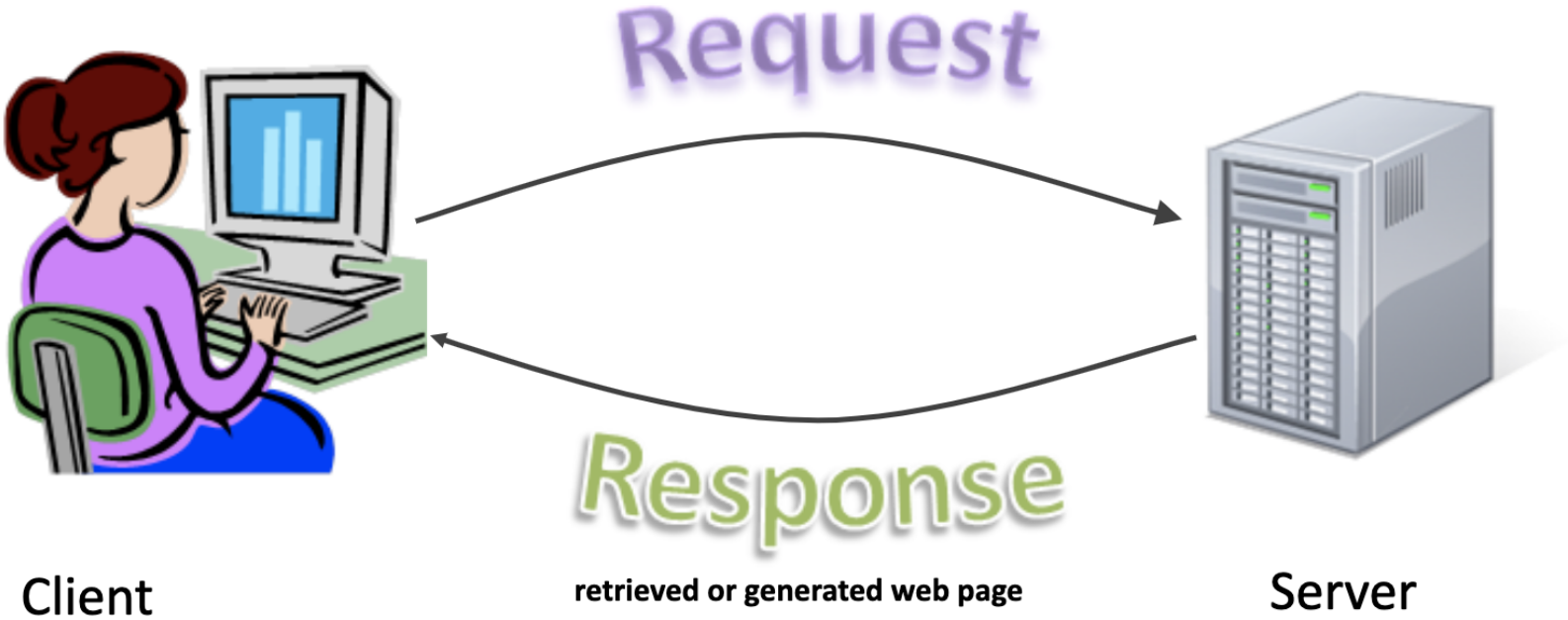
FULL STACK WEB DEVELOPMENT

www.spacode.co.il


Request
Response
Client
retrieved or generated web page
Server


FRONT END
DEVELOPMENT
< HTML >
< DIV >
JAVASCRIPT
CSS HTML


BACK END
DEVELOPMENT
<code>
</>
PYTHON PHP
RUBY JAVA GO

Full Stack

HTML Basics

DEFINITIONS

- WWW -- a software infrastructure layered on top of the Internet
- HTTP -- HyperText Transport Protocol, layered on top of TCP
- HTTPS -- secure HTTP using encryption
- HTML -- HyperText Markup Language, version 4.01 is current

HTML PAGE FORMAT

```
<HTML>
    <HEAD>
     <TITLE> Qi’s web! </TITLE>
    </HEAD>
    <BODY>
        <H1> Hello World </H1>
        <! Rest of page goes here. This is a comment. >
    </BODY>
</HTML>
```

BODY ELEMENT

<BODY attributename="attributevalue">

- Deprecated attributes (but still used)
  - BACKGROUND=“*Sunset.jpg”* (can be tiled)
  - BGCOLOR=color
  - TEXT=*color*
  - LINK=*color* (unvisited links)
  - VLINK=*color* (visited links)
  - ALINK=*color* (when selected)

HEADINGS

<H1 ...> text </H1> -- largest of the six
<H2 ...> text </H2>
<H3 ...> text </H3>
<H4 ...> text </H4>
<H5 ...> text </H5>
<H6 ...> text </H6> -- smallest of the six

ALIGN=*"position"* --left (default), center or right

HEADINGS

```
<HTML>
<HEAD>
  <TITLE>Document Headings</TITLE>
</HEAD>
<BODY>
Samples of the six heading types:
<H1>Level-1 (H1)</H1>
<H2 ALIGN="center">Level-2 (H2)</H2>
<H3><U>Level-3 (H3)</U></H3>
<H4 ALIGN="right">Level-4 (H4)</H4>
<H5>Level-5 (H5)</H5>
<H6>Level-6 (H6)</H6>
</BODY>
</HTML>
```

<P> ELEMENT

- <P> defines a paragraph
- Add ALIGN=*"position"* (left, center, right)
- Multiple <P>'s do not create blank lines
- Use <BR> for blank line
- Fully-specified text uses <P> and </P>
- But </P> is optional

<P> ELEMENT

```
<BODY>
<P>Here is some text </P>
<P ALIGN="center"> Centered text </P>
<P><P><P>
<P ALIGN="right"> Right-justified text
<! Note: no closing /P tag is not a problem>
</BODY>
```

SPECIAL CHARACTERS

Character	Use
<	<
>	>
&	&
"	"
Space	

COLORS

- Values for BGCOLOR and COLOR
  - many are predefined (red, blue, green, ...)
  - all colors can be specified as a six character hexadecimal value: RRGGBB
  - FF0000 – red
  - 888888 – gray
  - 004400 – dark green
  - FFFF00 – yellow

FONTS

```
<FONT COLOR="red" SIZE="2" FACE="Times Roman">
This is the text of line one </FONT>
<FONT COLOR="green" SIZE="4" FACE="Arial">
Line two contains this text </FONT>
<FONT COLOR="blue" SIZE="6" FACE="Courier"
The third line has this additional text </FONT>
```

ORDERED (NUMBERED) LISTS

```
<OL TYPE="1">
  <LI> Item one </LI>
  <LI> Item two </LI>
  <OL TYPE="I" >
    <LI> Sublist item one </LI>
    <LI> Sublist item two </LI>
    <OL TYPE="i">
      <LI> Sub-sublist item one </LI>
      <LI> Sub-sublist item two </LI>
    </OL>
  </OL>
</OL>
```

UNORDERED (BULLETED) LISTS

```
<UL TYPE="disc">
  <LI> One </LI>
  <LI> Two </LI>
  <UL TYPE="circle">
    <LI> Three </LI>
    <LI> Four </LI>
    <UL TYPE="square">
      <LI> Five </LI>
      <LI> Six </LI>
    </UL>
  </UL>
</UL>
```

<A> ELEMENT (HYPERLINK)

Link to an absolute URL:

If you get spam, contact <A HREF="htttp:www.microsoft.com"> Microsoft </A> to report the problem.

Link to a relative URL:

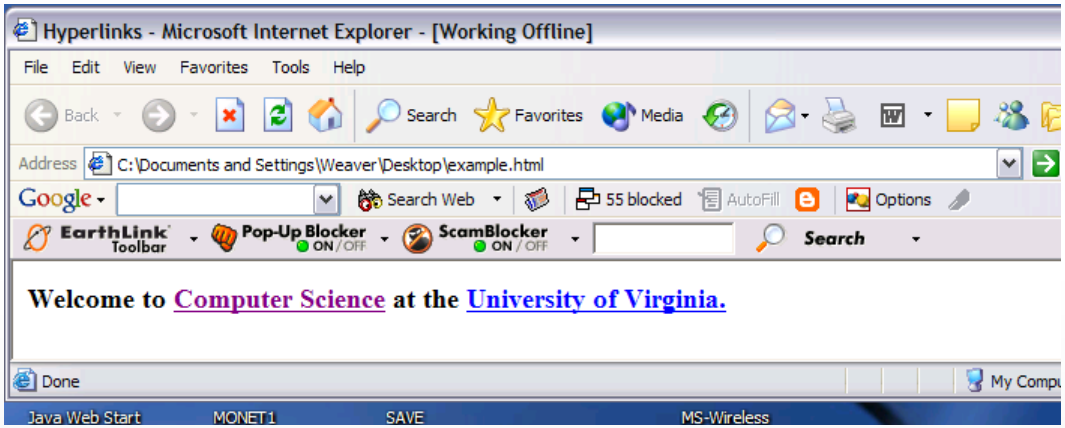
See these <A HREF="#references"> references </A> concerning our fine products.

Link to a section within a URL:

Amazon provided a <A HREF="www.amazon.com/#reference"> reference for our company. </A>

HYPERLINKS

```
<BODY>
<H3>Welcome to <A HREF="http://www.cs.virginia.edu">
<STRONG>Computer Science</STRONG></A>
at the <A HREF="www.virginia.edu">University of Virginia.</A>
</H3>
</BODY>
```



IMAGES

- SRC is required
- WIDTH, HEIGHT may be in units of pixels or percentage of page or frame
  - WIDTH="357"
  - HEIGHT="50%"
- Images scale to fit the space allowed

IMAGES

Align=*position*	Image/Text Placement
Left	Image on left edge; text flows to right of image
Right	Image on right edge; text flows to left
Top	Image is left; words align with top of image
Bottom	Image is left; words align with bottom of image
Middle	Words align with middle of image

TABLES

<TABLE>	table tag
<CAPTION>	optional table title
<TR>	table row
<TH>	table column header
<TD>	table data element

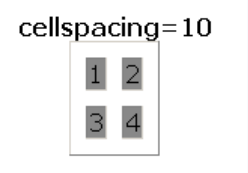
TABLES

```
<TABLE BORDER=1>
  <CAPTION>Table Caption</CAPTION>
  <TR><TH>Heading1</TH>        <TH>Heading2</TH></TR>
  <TR><TD>Row1 Col1 Data</TD><TD>Row1 Col2 Data</TD></TR>
  <TR><TD>Row2 Col1 Data</TD><TD>Row2 Col2 Data</TD></TR>
  <TR><TD>Row3 Col1 Data</TD><TD>Row3 Col2 Data</TD></TR>
</TABLE>
```

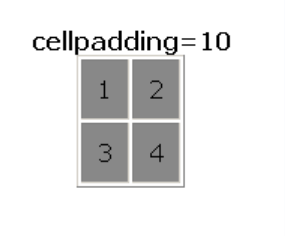
<TABLE> ELEMENT ATTRIBUTES

- ALIGN=*position* -- left, center, right for table
- BORDER=*number* -- width in pixels of border (including any cell spacing, default 0)
- CELLSPACING=*number* -- spacing in pixels between cells, default about 3
- CELLPADDING=*number* -- space in pixels between cell border and table element, default about 1
- WIDTH=*number[%]*-- width in pixels or percentage of page/frame width

- **cellspacing=10**



- **cellpadding=10**



<TABLE> ELEMENT ATTRIBUTES

BGCOLOR=*color* -- background color of table, also valid for <TR>, <TH>, and <TD>
RULES=*value* -- which internal lines are shown; values are none, rows, cols, and all (default)

EX: <TABLE COLS="40%, *,*">
<TABLE ROWS="*,*">

<TD> TABLE CELL ATTRIBUTES

Valid for the table cell:
colspan -- how many columns this cell occupies
rowspan – how many rows this cell occupies

```
<TABLE ALIGN="center" WIDTH="300" HEIGHT="200" border="1">
<TR>
<TD colspan="1" rowspan="2">a</TD>
<TD colspan="1" rowspan="1">b</TD>
</TR>
<TR>
<TD colspan="1" rowspan="1">c</TD>
</TR>
</TABLE>
```

FRAMES

- Frames help control navigation and display
- <FRAME> attributes include
  - FRAMEBORDER – yes or 1 for borders
  - FRAMESPACING – width of border
  - BORDERCOLOR – color
  - SRC – location of HTML to display in frame
  - NAME – destination for TARGET attribute

FRAMES

- MARGINWIDTH – left/right margins
- MARGINHEIGHT – top/bottom margins
- SCROLLING – yes or 1 adds scroll bar
- NORESIZE – yes or 1 disables resizing

FRAMES

```
<FRAMESET ROWS="75%,25%">

  <FRAMESET COLS="*,*,*">
    <FRAME SRC="http://www.virginia.edu">
    <FRAME SRC="http://www.virginia.edu">
    <FRAME SRC="http://www.virginia.edu">
  </FRAMESET>

  <FRAMESET COLS="*,*">
    <FRAME SRC="http://www.virginia.edu">
    <FRAME SRC="http://www.virginia.edu">
  </FRAMESET>

</FRAMESET>
```

FORMS

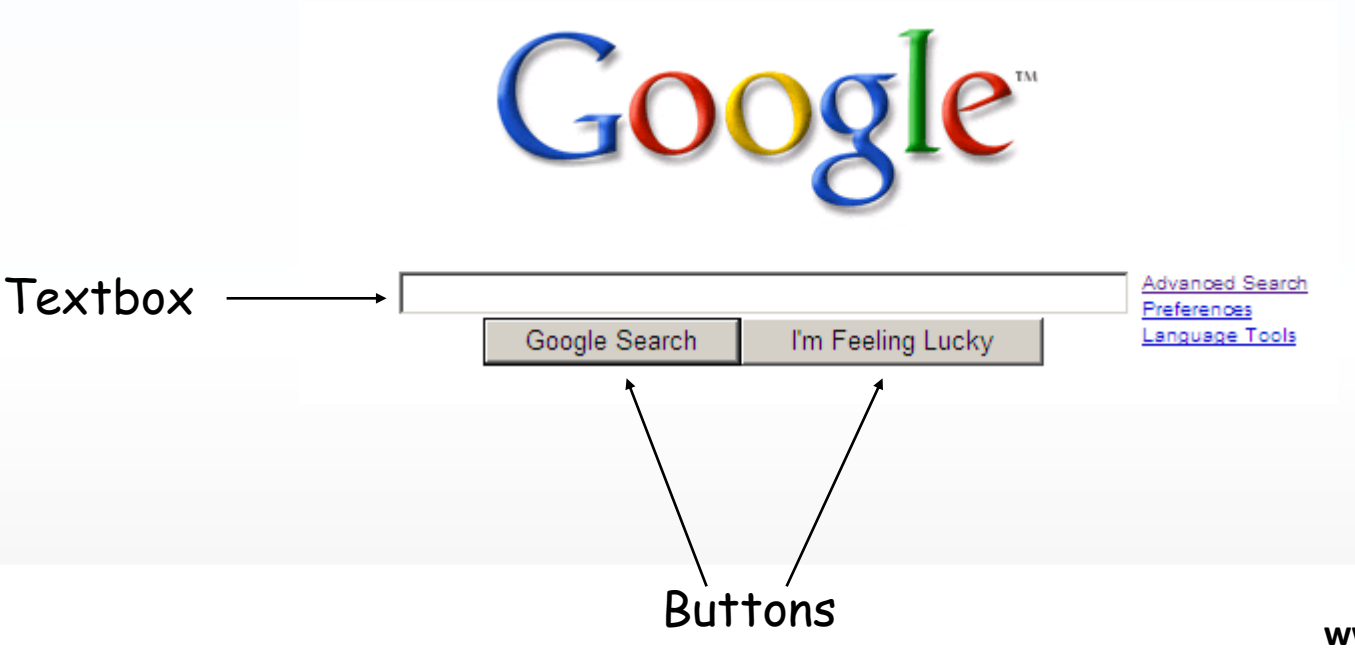
Forms are the easiest way to make web pages “interactive”.

– They allow the user to send some input data to the server for processing.

Like a paper-based form, a *Web form* allows the user to enter requested information and submit it for processing

FORMS

- For example, the *Google* web page has a textbox to allow users to enter search words, and clickable buttons to allow users to activate the search.



HTML forms may include different kinds of user input:

- text boxes
- text area
- check boxes
- menu choices
- etc.

A *Web form* is an HTML entity that can contain *form elements*.

Form elements are HTML elements that allow the user to enter information (in text fields, *textarea* fields, drop-down menus, radio buttons, checkboxes, etc.).

A *Web form* is defined with the HTML <form> element.

E.g.

```
<form  attributes>
   <input ... />
   <input ... />
   ...
</form>
```

Define the input type to the form.
Note there are other elements besides <input />

INPUT

The most used form tag is the <input> tag.

The actual type of input is specified with the type *attribute*.

The most commonly used input types are explained below.

Text Fields

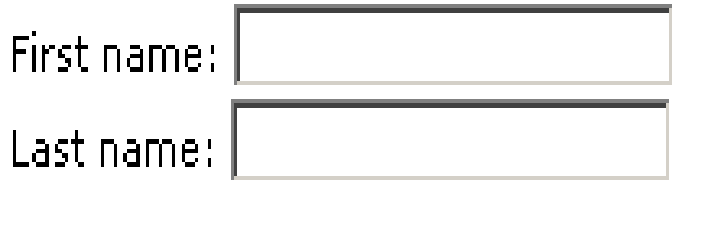
Text fields are used when you want the user to type a limited number of letters, numbers, etc. in a form box.

E.g. Google search box

Example:

```
<form>
    First name:
    <input type="text" name="firstname“/>
    <br />
    Last name:
    <input type="text" name="lastname“/>
</form>
```

How it looks in a browser:



The name attribute, e.g. firstname, lastname, allows the user to “name” (or associate) his/her input data with the given name.

If the browser user enters data into these two fields:

First name: John

Last name: Smith

This name attribute information is <u>sent to the server</u> along with the user input data, enabling software on the server to differentiate between the different inputs.

E.g. firstname=John & lastname=Smith

→ Sent to the server

Example

```
<form>
 Name:
 <input  type=“text"  name="Name"  size="35“/> <br />

 Address:
 <input  type=“text"  name="Address"  size="35“/> <br />

 E-mail:
 <input  type=“text"  name="E-mail"  size="35“/> <br />
</form>
```

Name:

Address:

E-mail:

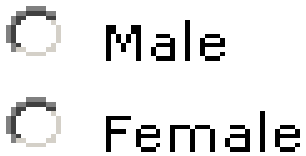
RADIO BUTTONS

Radio Buttons are used when you want the user to select one option from a limited number of choices.

```
<form>
   <input type="radio" name="gender" value="male"/> Male
   <br />
   <input type="radio" name="gender" value="female"/> Female
</form>
```

How it looks in a browser:

Note: name attribute has same value for each option



Note that only one option can be chosen. Each different option is specified with input element.

CHECKBOXES

Checkboxes are used when you want to allow the user to select <u>one or more</u> options of a limited number of choices.

```
<form>
    <input type="checkbox" name="bike“/>
      I have a bike
    <br />
    <input type="checkbox" name="car“/>
      I have a car
</form>
```

☐ I have a bike

☐ I have a car

Note: name attribute has different value for each option

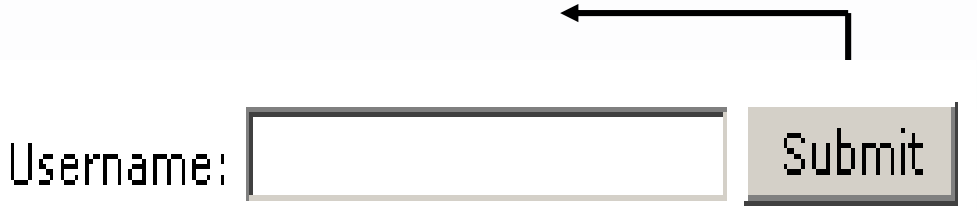
THE FORM'S ACTION ATTRIBUTE AND THE SUBMIT BUTTON

Define a “Submit” button to be used with your form.

When the user clicks on the “Submit” button, the input data content of the form is sent to a specified URL, e.g web server.

The action attribute of the <form> element defines the URL of the server that will receive and process the data content of the form.

```
<form   action="SomeURL"   method="post">
    Username:
    <input type="text" name="user"/>
    <input type="submit" value="Submit"/>
</form>
```



If instead of

```
<input type="submit" value="Submit“/>
```

we just have

```
<input type="submit“/>
```

i.e. no value attribute, then we get default submit text

May also generate a *Reset button*, which allows user to clear form and re-start all over again.

```
<input type="reset"/>
```

It may also take a value attribute.

TEXTAREA

A *textarea* is a <u>multi-line</u> text input box.

A user can write text in the text area.

In a text area you can write an unlimited number of characters.

```
<textarea rows="10" cols="30“>
      some optional initial text
</textarea>
```

BUTTONS

Define clickable buttons

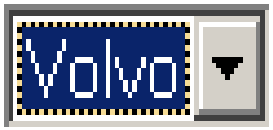
```
<form>
    <input type="button" value="Click“/>
</form>
```

DROP-DOWN MENUS

`<select>` tag defines menu or scrolling list of items.
`<option>` tag names each item

```
<form>
    <select name="car">
        <option value="volvo">Volvo</option>
        <option value="saab">Saab</option>
        <option value="fiat">Fiat</option>
        <option value="audi">Audi</option>
    </select>
</form>
```

Menu items



The default format for the transmission of form input data from the browser <u>to the server</u> is a concatenation of names and values, separated by “&”

element1_name=element1_value&element2_name=element2_value ...

```
<form   action=“SomeURL"   method=“post">
    <input type="text" name=“firstname“/>
    <input type="text" name=“lastname“/>
    <input type="submit" value="Submit“/>
</form>
```

firstname=John&lastname=Smith

FORM TAG

The <form> tag is used to construct interactive forms that can be used to allow user to "log in" or register; submit requests or order services; or just provide feedback and comments; …

Form Tag Attributes

action - the URL of the program/script used to process the input data

method - how the input data will be sent to the server

- GET or POST - preferred HTTP methods

name - an optional attribute used for reference

```
<form action=“some url" method=post>

        The form elements go here

</form>
```

CSS

Cascading Style Sheet

WHAT IS CSS?

CSS stands for ***Cascading Style Sheet***. Typical CSS file is a text file with an extention.**.css** and contains a series of commands or rules. **These rules tell the HTML how to display.**

*To create a style sheet, create a file using Notepad (PC) or Text Edit (Mac), save it as a .css document and start writing the CSS code (see right).

Style.css

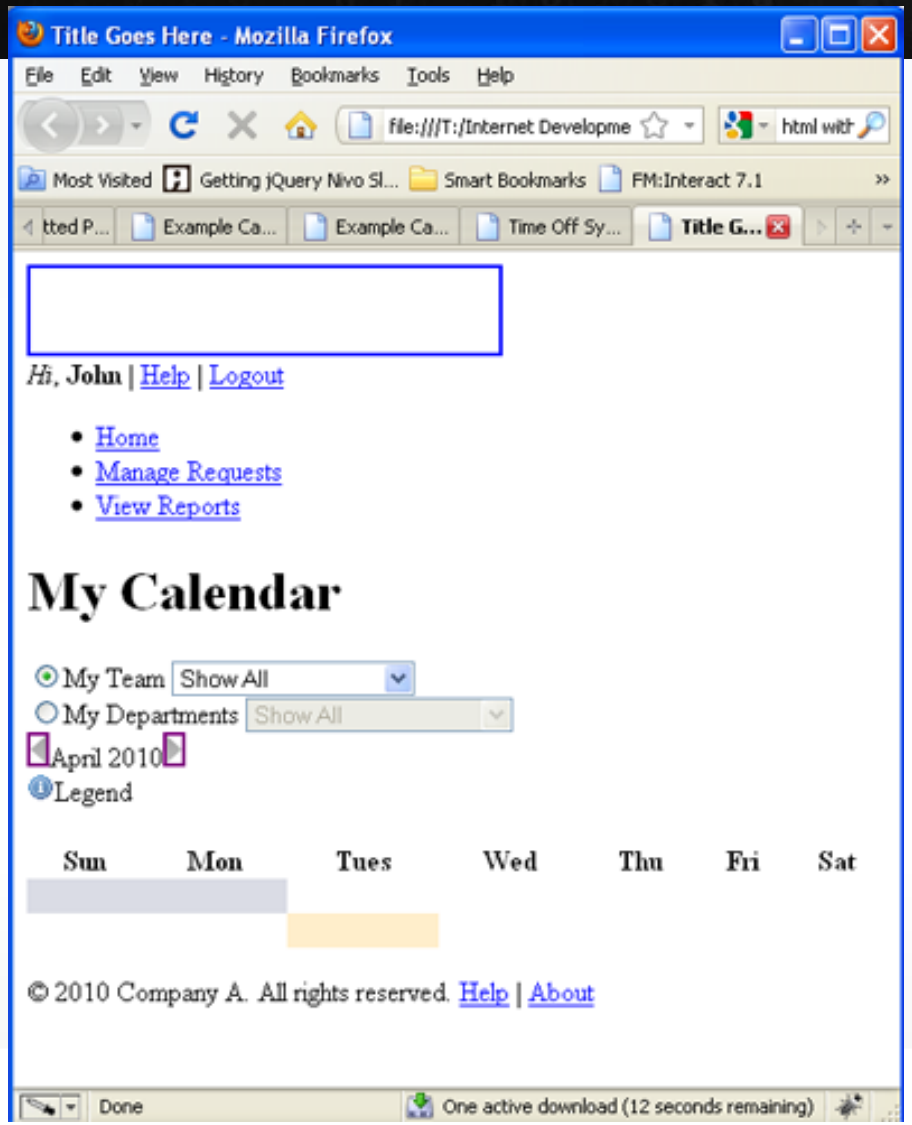
```
/* Styles for sitename.com*/

body {
font-family:Arial;
background: #000;
}

#container {
text-align:left;
width:1020px;

}

#header {
height:232px;
}

#footer {
width: 100%;
padding: 0 10px;
margin-bottom: 10px;

}
```

And so on….

CSS BENEFITS

- Separates structure from presentation
- Provides advanced control of presentation
- Easy maintenance of multiple pages
- Faster page loading
- Better accessibility for disabled users
- Easy to learn

HTML WITHOUT CSS



"HTML without CSS is like a piece of candy without a pretty wrapper."

Without CSS, HTML elements typically flow from top to bottom of the page and position themselves to the left by default.

With CSS help, we can create containers or DIVs to better organize content and make a Web page visually appealing.

HTML & CSS

- HTML and CSS work together to produce beautiful and functional Web sites
- HTML = structure
- CSS = *style*

ATTACHING A STYLE SHEET

Attach a style sheet to a page by adding the code to the <head> section of the HTML page. There are ***3 ways*** to attach CSS to a page:

1. External Style Sheet: Best used to control styling on multiple pages.

```
<link rel="stylesheet" type="text/css"
media="all" href="css/styles.css" />
```

2. Internal Style Sheet: Best used to control styling on one page.

```
<style type="text/css">
h1 {color: red)
</style>
```

3. Inline Style Sheet*: CSS is not attached in the <header> but is used directly within HTML tags.

```
<p style="color: red">Some Text</p>
```

CSS RULE STRUCTURE

A CSS RULE is made up of a selector and a declaration. A declaration consists of property and value.

```
selector {property: value;}
```

declaration

SELECTORS

A selector, here in green, is often an element of HTML.

```
body { property: value; }
h1 { property: value; }
em { property: value; }
p { property: value; }
```

PROPERTIES AND VALUES

```
body {background: purple;}
h1 {color: green; }
h2 {font-size: large;}
p {color: #ff0000;} /*hexadecimal for
  red*/
```

Properties and values tell an HTML element how to display.

```
body {
background: purple;
color: green;
}
```

*CSS code can be written in a linear format (above) or in a block format (below).

GROUPING SELECTORS

Group **the same selector** with different declarations together on one line.

```
h1 {color: black;}
h1 {font-weight: bold;}
h1 {background: white;}
```

Example of grouping selectors (both are correct):

```
h1 {
color: black;
font-weight: bold;
background: white;
}
```

GROUPING SELECTORS

Group **different selectors** with the same declaration on one line.

```
h1 {color: yellow;}
h2 {color: yellow;}
h3 {color: yellow;}
```

Example of grouping selectors ***(both are correct)****:*

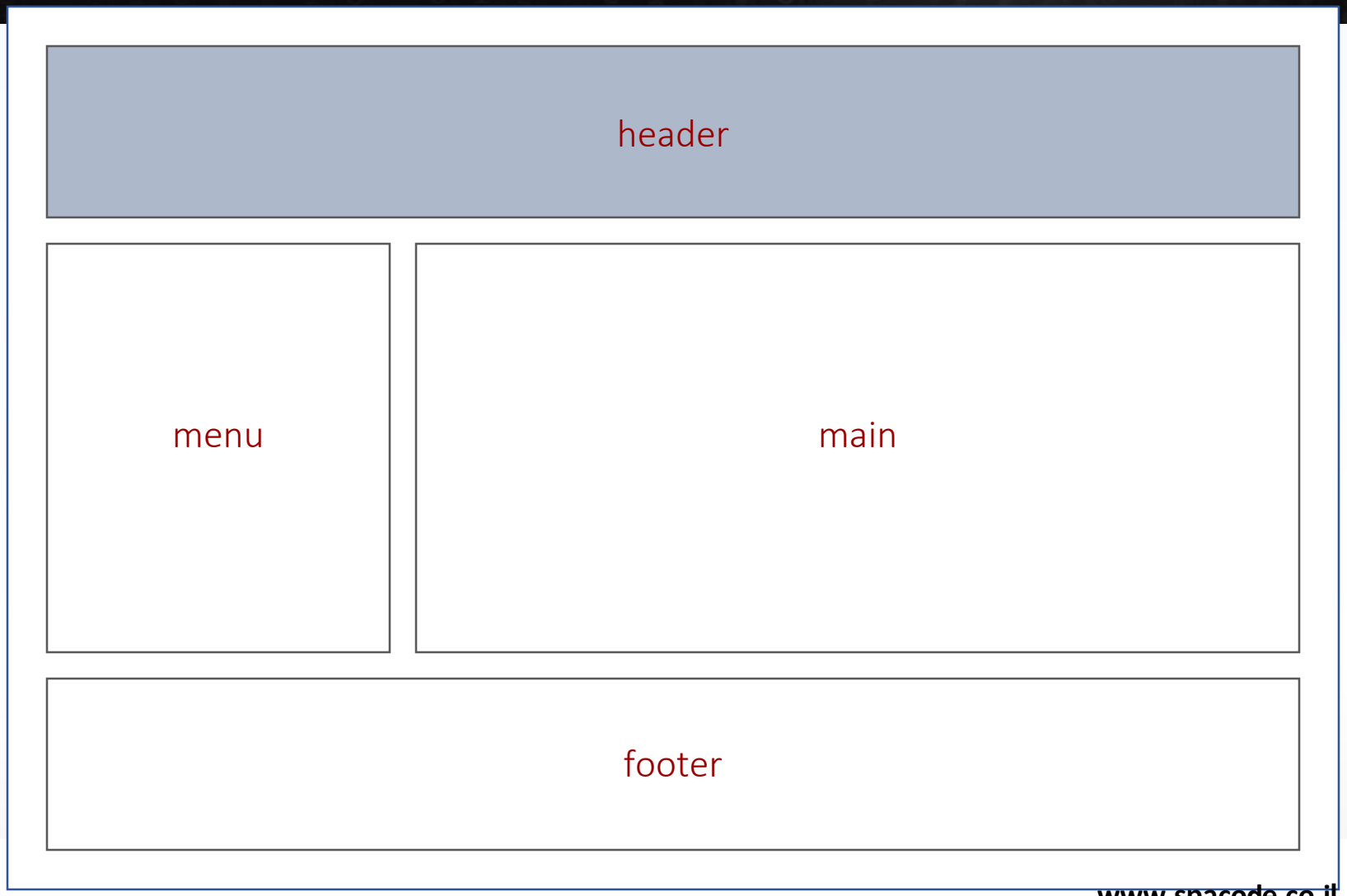
```
h1, h2, h3 {color: yellow;}
```

COMMENTS IN CSS

- Explain the purpose of the coding
- Help others read and understand the code
- Serve as a reminder to you for what it all means
- Starts with /*and ends with*/

```
p {color: #ff0000;} /*Company Branding*/
```

TYPICAL WEB PAGE (BROWSER)



TYPICAL WEB PAGE (HTML)

Typical HTML Web page is made up of containers (boxes) or DIVs. Each DIV is assigned an ID or a Class.

```
<div id="container">
    <div id="header">Insert Title</div>
    <div id="main">content
        <div id="menu">content</div>
    </div>
    <div id="footer">content</div>
</div>
```

TYPICAL WEB PAGE (CSS)

The CSS file uses the same DIV/ID/Class names as the HTML and uses them to style the elements.

```
#container {property: value;}

#menu {property: value;}

#main {property: value;}

#footer {property: value;}
```

IDS AND CLASSES

- **IDs (#)** are unique and can only be used once on the page
- **Classes (.)** can be used as many times as needed

HTML Code:

```
<h1 id="mainHeading">Names</h1>
<p class="name">Joe</p>
```

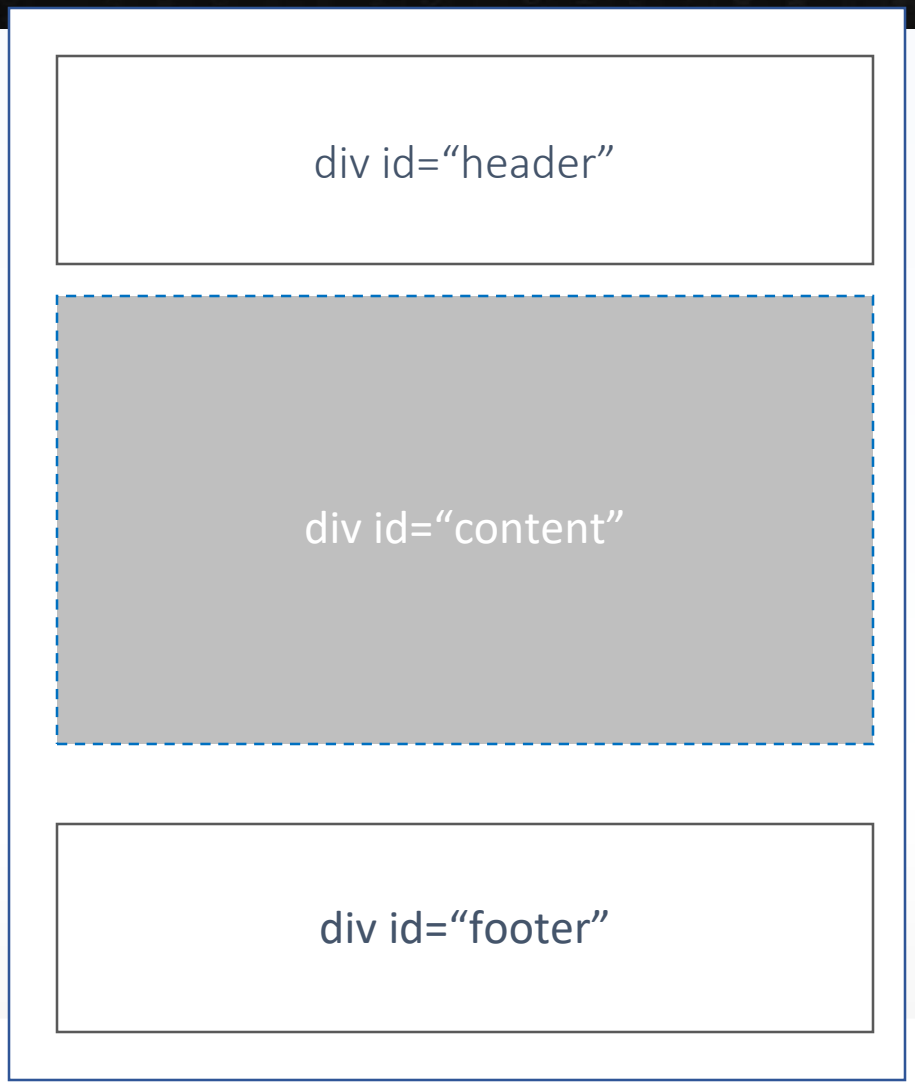
CSS Code:

```
#mainHeading {color: green}
.name {color: red}
```

CSS BOX PROPERTIES

- Background-color
- Width
- Padding
- Margin
- Border-width
- Border-color
- Border-style

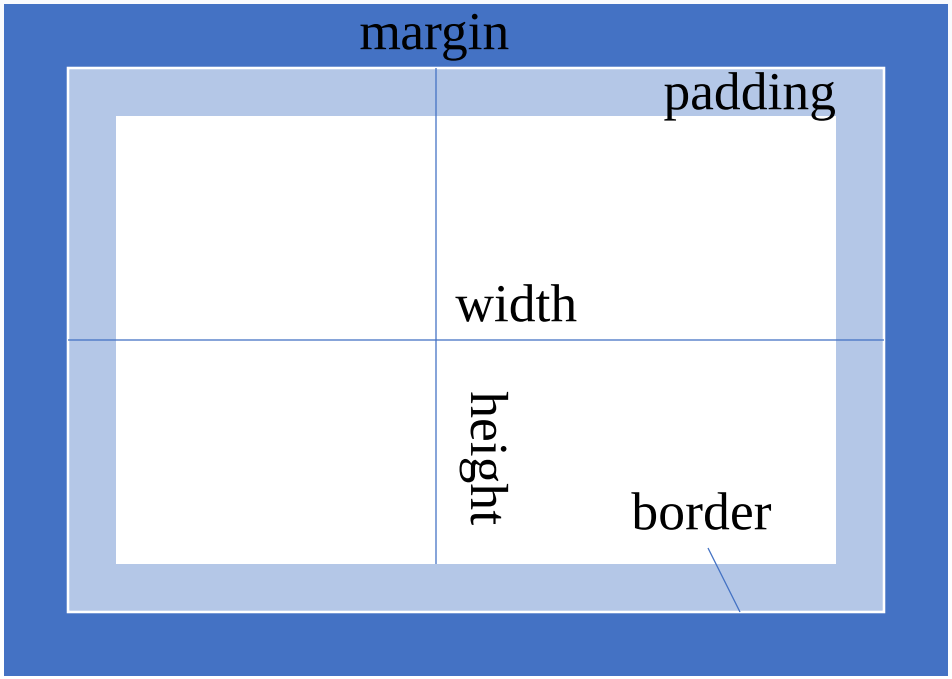
CSS HTML




```
#content {
background-color: #ccc;
margin-bottom: 10px;
border: 1px dashed blue;
color: #fff;
width: auto;
}
```

COMMON CSS LAYOUT PROPERTIES

- Width
- Height
- Float
- Clear
- Border
- Padding
- Margin



WIDTH & HEIGHT

Width and height define the width and height of an element.

```
div id="box"
```

```
#box {width="50px"}
#box {width="50em"}
#box {width="100%"}
#box {width="auto"}
```

```
#box {height="auto"}
```

*Width and height can be specified in pixels, ems, percentages or set to auto

TEXT PROPERTIES

MAIN HEADING

Gravida lacinia velit. Vivamus tortor enim, tincidunt at, pellentesque ut, iaculis eu, quam.

```
.mainHeading {
color: red;
letter-spacing: 5px;
text-transform: uppercase;
word-spacing: 15px;
text-align: left;
font-family: Times;
text-decoration: underline;
font-size: 12px;
font-style: italic;
font-weight: bold;
}
```

To style the main heading in the paragraph above, we assigned a class the HTML tag.

```
<h3 class="mainHeading">Main Heading</h3>
```

CSS COLORS

Standard

- White
- Black
- Blue
- Fuchsia
- Gray
- Green
- Lime
- Aqua

Hexadecimal

- #ffffff
- #fff
- #cccf0f3

STYLING LINKS

The links property defines how inactive, hovered, active, and visited link states appear to the user.

```
a:link {color: red; text-decoration: none;border-bottom: 1px dashed red; background: white;}

a:visited {color: yellow;}

a:active {color: green;}

a:hover {color: orange;}
```

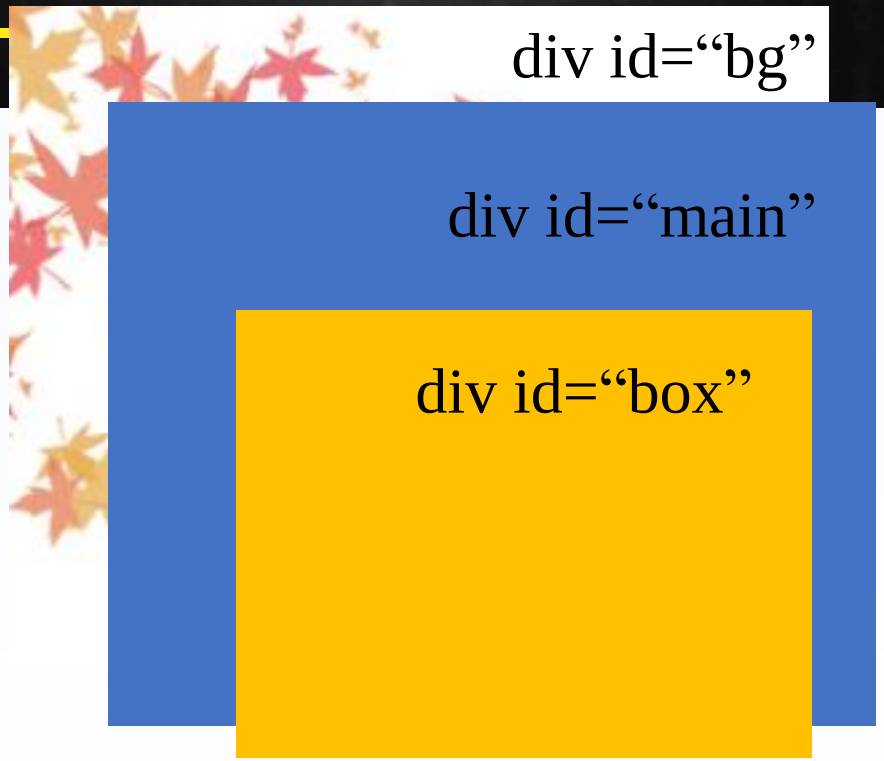
INCLUDING IMAGES

Properties for working with images include:

- Background-image
- Background-repeat
- Background-position
- Background-attachment

LAYERING

Background colors and images are layered like sheets of paper one on top of the other.




```
#bg {background:url(leaves.jpg) no-repeat top left}
#main {background-color: red}
#box {background-color: yellow}
```

BACKGROUND-IMAGE

The background-image property sets an image in the background of an element.

- Background images and colors are layered.
- If not transparent, the last one listed in the CSS file is visible.

```
li {
background-image:url(flower.jpg);
padding-left: 10px;
}
```

BACKGROUND-REPEAT

The background-repeat property sets an image in the background of an element and tiles, or repeats, it. Tiling is the default.

```
li {
background-image:url(flower.jpg);
background-repeat:no-repeat;
}
```

Possible Values >

- repeat
- repeat-x (horizontal)
- repeat-y (vertical)
- no-repeat

IMAGE POSITIONING

The background-position property positions the image using either combined keywords (top, bottom, left, right, and center); length values; or percentage values.

```
background-position: right top;
/*can also use number values*/

background-attachment: fixed;
/*can also use 'scroll'*/
```

left top

center top

right bottom

left bottom

center bottom

The background-attachment property fixes or scrolls an image in the browser window. Values include *fixed* and *scroll*.

THE POWER OF CASCADE

When multiple styles or style sheets are used, they start to cascade and sometimes compete with one another due to CSS's inheritance feature. Any tag on the page could potentially be affected by any of the tags surrounded by it.

So, which one wins? Nearest Ancestor Wins.

1. Inline style or directly applied style
2. The last style sheet declared in the <header> section

SAVING TIME WITH INHERITANCE

In a nutshell, **inheritance** (not the money you get from your grandma) is the process by which CSS properties applied to one tag are passed on to nested tags.

For example, the paragraph tag will inherit the same styling as the body tag because <p> is always located inside <body>.

```
<body style="font-family: Arial">
   <p>This text will be Arial as well</p>
</body>
```

So, instead of styling each paragraph separately, you can define the font color in the <body>, and everything inside will have that color.